Investigating Forward kinematic Analysis of a 5-axes Robotic Manipulator Using Denavit-Hartenberg Method and Artificial Neural Network

Israa R. Shareef
Al Khawarzmi Engineering College, University of Baghdad / Baghdad
Email : engineer_israa@hotmail.com
Iman A. Zayer
Al Khawarzmi Engineering College, University of Baghdad / Baghdad
Izzat A. Abd Al Kareem
Al Khawarzmi Engineering College, University of Baghdad / Baghdad

Received on: 18/12/2013 & Accepted on: 7/8/2014

ABSTRACT

Robot Forward kinematic equations' analysis is an essential and important manner to analysis the position and orientation of the end effectors of a robotic manipulator, where in this paper, Denavit-Hartenberg notation and method (D – H) is used to represent the relative kinematic relationships precisely between each two adjacent links of this robot , besides a kind of artificial neural network (ANN) which is known as the supervised learning training sets network is investigated to solve the problem of kinematic analysis of this laboratory five - axes robot, it shows that the using of the artificial intelligence method which is the neural networks had offered the facility of dealing with this non linear robotic system by a simple manner with acceptable faster solution as compared with the traditional forward kinematic equations analysis method .

Keywords : Denavit–Hartenberg (D – H) method , Artificial Neural Network (ANN) , 5-DOF (Degree Of Freedom) robotic manipulator .

استقصاء التحليل الحركي الأمامي لإنسان آلي مناول ذو خمسة محاور باستخدام طريقة دينافيت – هارتنبيرغ و الشبكة العصبية الصناعية.

الخلاصة

إن تحليل المعادلات الحركية الأمامية للإنسان الآلي هو أسلوب أساسي و مهم لتحليل موقع و اتجاه النهاية المؤثرة للإنسان الآلي المناول , حيث انه في هذه الدراسة تم استخدام ترميز و طريقة دينافيت – هارتنبيرغ لتمثيل العلاقات الحركية النسبية بين كل وصلتين متجاورتين للإنسان الآلي بدقة . إلى جانب ذلك , تم الأخذ بأحد أنواع الشبكات العصبية الصناعية و التي تعرف بشبكة مجموعة المحاولات التعليمية المشرف عليها , لحل مشكلة التحليل الحركي لهذا الإنسان الآلي ألمختبري ذو الخمس درجات لحرية الحركة , و ظهر أن استخدام طريقة الذكاء الاصطناعي متمثلا

بالشبكات العصبية وفرت إمكانية التعامل مع منظومة الإنسان الآلي اللاخطية باسلوب مبسط و حل مقبول و سريع مقارنة بطريقة التحليل التقليدي للمعادلات الحركية الأمامية .

INTRODUCTION

Robotic manipulator is a collection of links that connect to each other by joints, these joints can be revolute or prismatic , where revolute joint has a rotary motion around an axis and prismatic joint has a linear motion around an axis. Each joint provides one or more degrees of freedom (DOF) [1] , in this paper , we will discuss the motion of a 5 – DOF robotic manipulator which is shown in figure (1) in its two views (this robot is designed & manufactured in laboratory) , with revolute joints that have (Base , Shoulder , Elbow , Tool Roll , and Tool yaw) rotation motions .

Generally , Kinematic studies the motion of bodies without consideration of the forces or moments that cause this motion , while robot kinematics refer to the analytical study of the motion of a robot manipulator. Formulating the suitable kinematics models for a robot mechanism is very crucial for analyzing the behavior of industrial manipulators [2] .

In fact the manipulator kinematics is a study of the geometry of manipulator arm motions. Since the performance of specific manipulator tasks is achieved through the movement of the manipulator linkages, kinematic is of fundamental importance in robot design and control besides, the kinematic equation provides the relationship between the joint displacement and the resulting end-effector position and orientation.

The problem of finding the end-effector position and orientation for a given set of joint displacements is referred to as the forward kinematics problem [3]. That is, the forward kinematics problem allows one to specify in a unique manner the relationship between the joint vector (θ) and the Cartesian vector (*x)* as:

$$x(t) = f(\theta(t))$$

Where f is the function defining the forward kinematic relation of the manipulator, and t is the time. Normally, the forward Kinematic equation can be obtained either classically from the spatial geometry of the manipulator or by solving certain matrix algebraic equations, or from an artificial intelligence method like the neural network as what will be discussed in this paper .

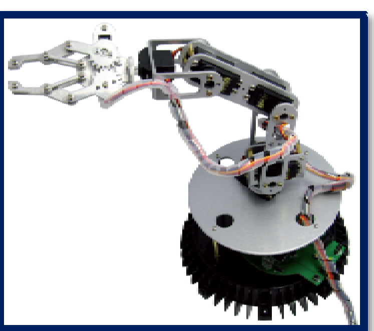



Figure (1) : The 5- DOF robotic manipulator

DENAVIT–HARTENBERG ANALYSIS

Kinematics analysis starts with the determination of the position of each joint given the configuration of the robot, the base position and orientation, the angles of the revolute joints, and the length of the links of the robot; this is called the *forward kinematics solution* (FKS) of the robot [4].

Denavit & Hartenberg (1955) showed that a general transformation between two joints requires four parameters. These parameters known as the Denavit-Hartenberg (DH) parameters have become the standard for describing robot kinematics [2].

In order to represent the relative kinematics relationship precisely between each two adjacent links, the ***Denavit-Hartenberg*** (D-H) notation will be followed in this paper.

The four important (DH parameters) as was defined in [5] that relate neighboring coordinate systems are :

• α_i is the angle from z_{i-1} to z_i, measured about x_i.
• d_i is the distance from x_{i-1} to x_i, as measured along z_{i-1}.
• a_i is the distance from z_{i-1} to z_i measured along x_i.
• θ_i is the angle from x_{i-1} to x_i, measured about z_{i-1} [6].

To calculate the position of the end effector in the universe frame , a series of matrix calculations must be performed which find the position of the end effector relative to each robotic joint , this series of matrix multiplications gives the position and orientation of the end-effector (the tool) in the universe frame. Each rotational joint has a specific “joint angle” which is the angle of rotation of the joint axis from its home position. Each joint angle is designated using the Greek letter θ followed by the joint number.

DH formula for calculating forward kinematics (T_{i-1}^{i}) is :

$$\begin{bmatrix} \cos\theta_i & -\cos\alpha_i\cdot\sin\theta_i & \sin\alpha_i\cdot\sin\theta_i & a_i\cdot\cos\theta_i \\ \sin\theta_i & \cos\alpha_i\cdot\cos\theta_i & -\sin\alpha_i\cdot\cos\theta_i & a_i\cdot\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad ...(1)$$

Defining the D-H parameters as presented and writing the homogenous transformation matrices for i= 1… n (n is DOF of the robot) , represents the transformation of the body i or frame i+1 with respect to its previous body i-1 or the frame attached to it i.e. frame i The homogeneous transformation matrix of the end effector frame with respect to frame1 i.e. is now obtained by post-multiplication of the above individual homogeneous transformation Ti, for i=1…..n [7].

Artificial Neural Network

Interest in artificial intelligence control researches like neural networks has been increased lastly , to reduce the computational complexity of motion planning and control for manipulators and to deal with the non linear systems that would be difficult or impossible to model mathematically [8,9].

Besides , The availability of process control computers and associated data historians make it easy to generate neural network solutions for process modeling and control. Neural network solutions are well accepted in process systems since they are cost-effective, easy-to-understand, nonlinear (especially the multi layered neural networks), data-driven , and parallel distributed structure with learning ability [10,11].

Many of the neural networks for robot kinematic control are feed forward networks such as the multilayer perceptron trained via supervised learning using the back propagation algorithm or its variants. Recurrent neural networks with feedback connections, such as the Hopfield networks, have also been applied for kinematic control[8].A supervised teach NN is used in this paper as a learning rule .

Kinematic Modeling of the Robot

The laboratory robot which is used in this paper is a 5-axes manipulator composed of serial links which are affixed to each other by revolute joints, each joint is a servo motor from the base to the end effector. The forward kinematic model is obtained using the previously mentioned (D-H) method. First of all, the coordinate frames are attached to each joint of the robot as shown in (figure 2) , according to the assigned frames the parameters of the DH for the robot a, α, d, and θ which represent for the robot ; link length, link twist, link offset, and joint angle respectively which is shown in (table 1) .

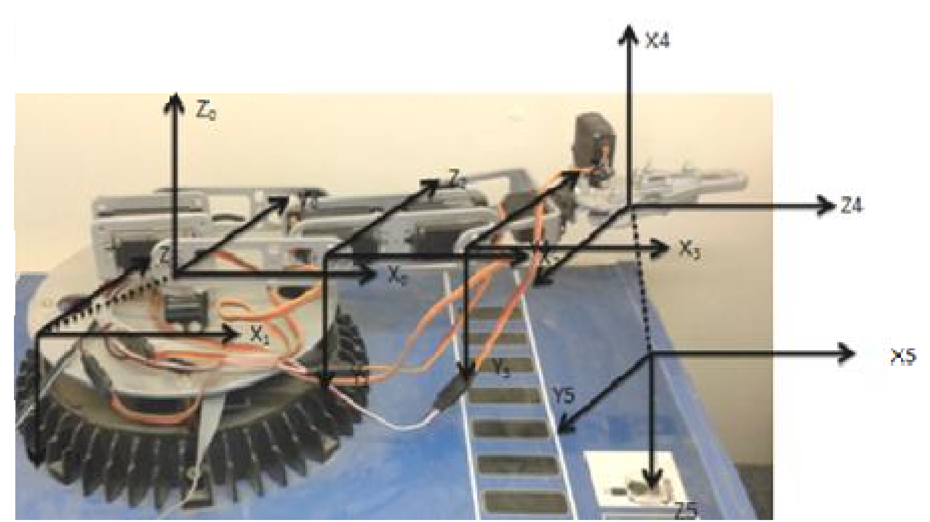


Figure (2) : The joints' frames of the robot

Table 1: D-H parameters of the robot

Axis	θ	d	a	α	Home
1	θ_1	0	0	-π/2	0
2	θ_2	0	8	0	0
3	θ_3	0	8	0	0
4	θ_4	0	0	- π/2	-π/2
5	θ_5	1.4	0	0	0

Where angles are measured in degrees , lengths are measured in centimeters , and Home is the default joint angle of the robot .

In order to get the position of the end effector i.e. the coordinates x, y, z ; the link transformation matrix (eq. 1) is used. The mathematic representation of a kinematic chain is

$$T_{BASE}^{TOOL} = T_{BASE}^{WRIST} . T_{WRIST}^{TOOL} \qquad \ldots (2)$$

Forward kinematic $T_{BASE}^{WRIST} = T_0^1 T_1^2 T_2^3$ …(3)

$$= \begin{bmatrix} C_1 & 0 & -S_1 & 0 \\ S_1 & 0 & C_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_2 & -S_2 & 0 & a_2C_2 \\ S_2 & C_2 & 0 & a_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_3 & -S_3 & 0 & a_3C_3 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & C_1(a_2C_2 + a_3C_{23}) \\ S_1C_{23} & -S_1S_{23} & C_1 & S_1(a_2C_2 + a_3C_{23}) \\ -S_{23} & C_{23} & 0 & -(a_2S_2 + a_3S_{23}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (4)$$

$T_{WRIST}^{TOOL} = T_3^4 T_4^5$ …(5)

$$= \begin{bmatrix} C_4 & 0 & -S_4 & 0 \\ S_4 & 0 & C_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_5 & -S_5 & 0 & 0 \\ S_5 & C_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} C_4C_5 & -C_4S_5 & -S_4 & -d_5S_4 \\ S_4C_5 & -S_4S_5 & C_4 & d_5C_4 \\ -S_5 & -C_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad ...(6)$$

Then , we find T_{BASE}^{TOOL} to get the tool coordinates with respect to the base of the robot :

$$\mathbf{X} = -d_5\ C_1\ S_{234} + C_1\ [a_2C_2 + a_3C_{23}] \quad ...(7)$$

$$\mathbf{Y} = -d_5\ S_1\ S_{234} + S_1\ [a_2C_2 + a_3C_{23}] \quad ...(8)$$

$$\mathbf{z} = d_5\ C_{234} - [a_2S_2 + a_3S_{23}] \quad ...(9)$$

$where\ d_5 = 1.4\ cm,\ a_2 = a_3 = 8\ cm$

Neural Network Results and Discussions

In this paper , a supervised teach NN is used as a learning rule in which the input to the network is the interred set of five joints' angles of the robot , while the network target is the end-effector coordinates (x, y, z) as represented in equations 7 to 9 .

The neural output results are compared to the targets, the perceptron learning rule will then adjust and modify the weights and the biases of the network in order to move the network outputs closer to the targets. The number of layers are chosen to be five and the number of neurons is ten while the transfer function is Log-Sigmoid as below:

$$a = f(w_p + b) \quad ...(10)$$

$$a = \frac{1}{1+e^{-n}} \quad ...(11), \qquad n = w_p + b \quad ...(12)$$

Where n is the summer output , p is the scalar input , w is the scalar weight to form w_p.

b is bias, a is the scalar neuron output . The scalar input p represents x, y, z as follows:

$$x = f1(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \quad ...(13),$$

$$y = f2(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \quad ...(14),$$

$$z = f3(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \quad ...(15)$$

Where these equations are clearly accorded with equations 7, 8 & 9 respectively .

The neural network toolbox in MATLAB is used to investigate the supervised learning algorithm , where the neural network parameters are :

Network type is the feed forward back-propagation

Number of layers L=5 , Number of neurons K=10

The input vector n has five elements theta1, theta2, theta3 , theta4 and theta5

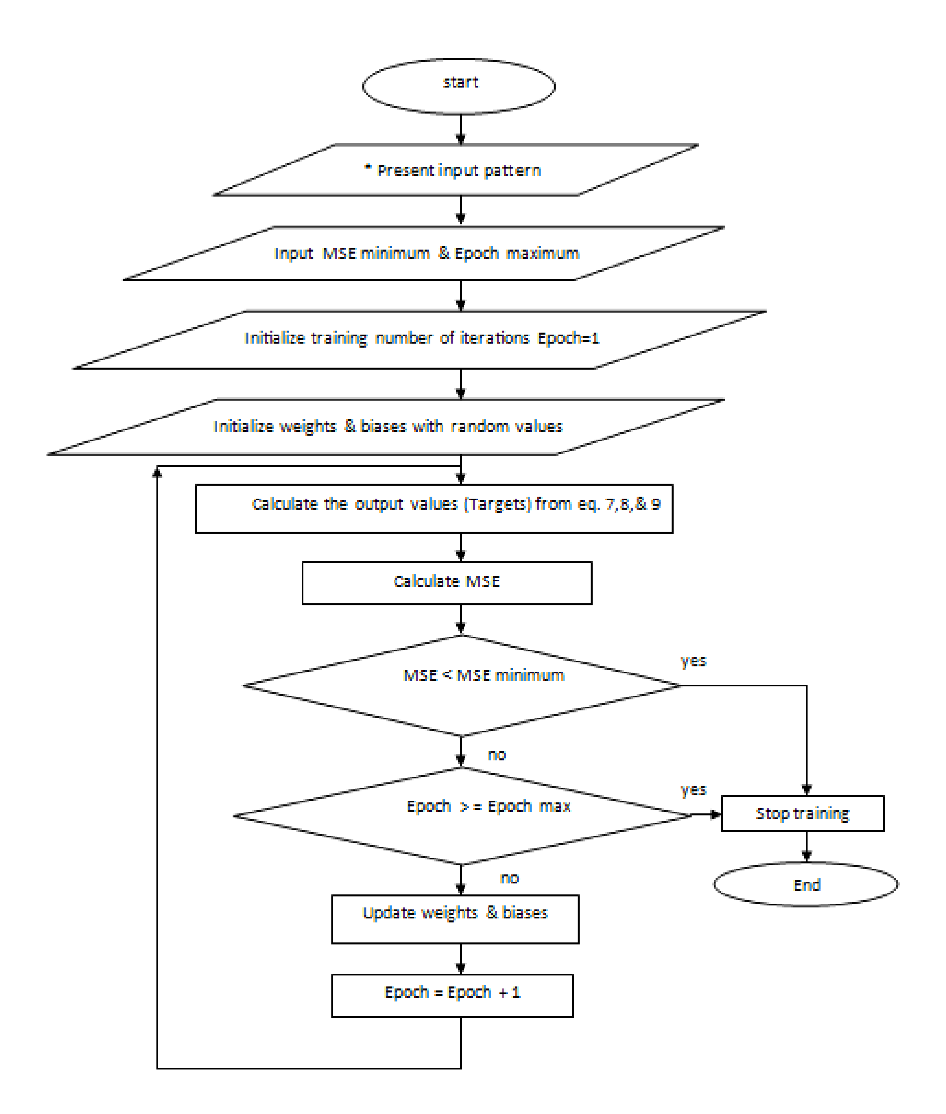
The output vector a has the three elements x, y, and z

The input weight matrix of the first layer WK,n

The second weight matrix Wa,K , Training function= trainlm

Adaptation learning function=learngdm , Performance function=mse (mean square error) , Transfer function= logsig.

The proposed neural network training process algorithm could be seen through the flow chart shown in figure (3) .



* : network type , training , adaptation learning , performance and transfer functions , L , K , n , a , Wk,n , Wa,k .

Figure (3) : The Training Process Flowchart

In figure (4) , the MATLAB neural network interface shows the learning rule progress and performance .

During neural network training , the progress is constantly updated in the training window. Of most interest are the performance, the magnitude of the gradient of performance and the number of validation checks. The magnitude of the gradient and the number of validation checks are used to terminate our NN training.

We notice that the gradient will become very small as the training reaches a minimum of the performance , while the network training will stop as the magnitude of the gradient is less than a limited number which can be adjusted by setting the min. gradient of the net .

The number of validation checks represents the number of successive iterations that the validation performance fails to decrease , the training will stop according to the chosen number of validation checks , where this criterion could be changed by setting the parameter max. fail of the net.

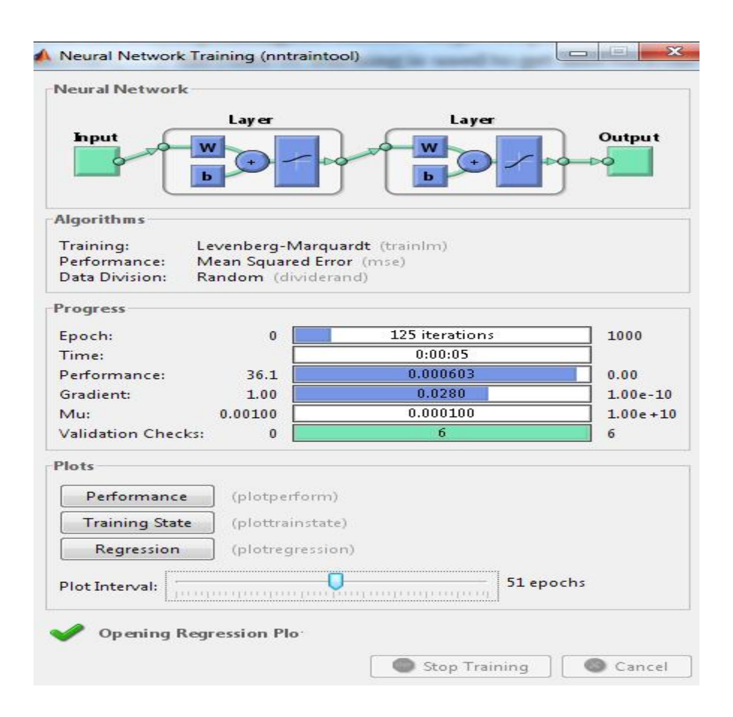


Figure (4) : The Neural Network Training Interface

From the training window, we can access the states of the NN performance, training , error histogram and regression. The performance plot as in figure (5) shows the value of the mean square error (which is the difference between the target & output values) versus the iteration number. It plots training, validation and test performances, we notice that the best validation performance 0.002615 had been achieved at epoch 119 . Besides , the mean square error had decreased in an obvious manner , so good NN training was accomplished .

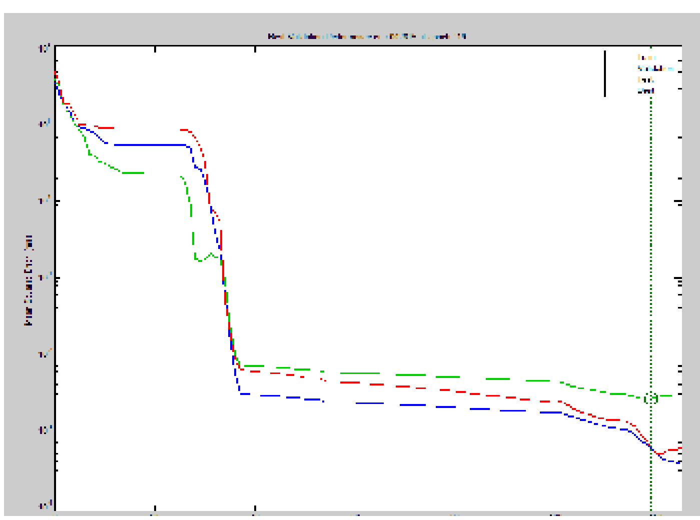


Figure (5) : The Neural Network Performance plot

In Figure (6) , we can see various gradients at epochs with the vises error percentage , the effect of Mu factor changing (which is the constant that had set at the beginning of the program , it controls how much the weights will be changed at each iteration) , and this figure also shows the failure situations during NN training .

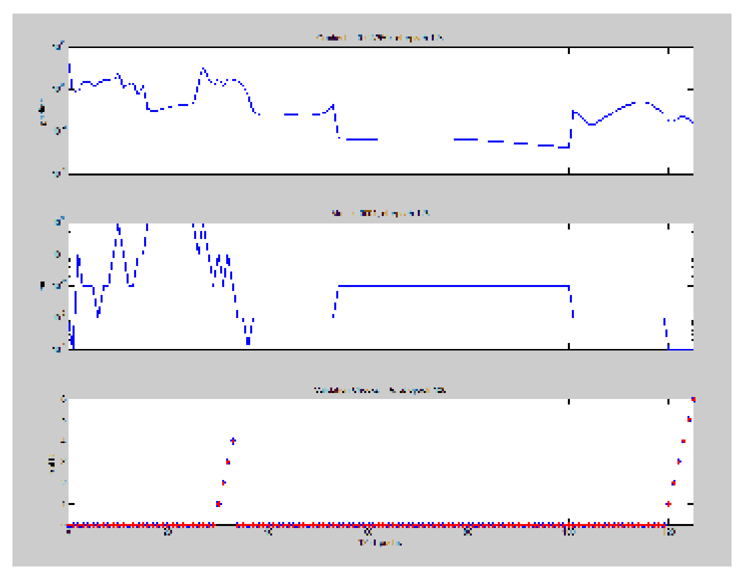


Figure (6) : Training state of the neural network

The regression plots show a regression between network outputs and network targets where we can use these regression plots to validate the network

performance and as well known in NNs , it's a good indicant if the target and the output are quietly convergent (making the factor R which relates each of the target & output is near from 1) , where this is exactly what could be shown in our suggested network as seen by plots of figure (7) .

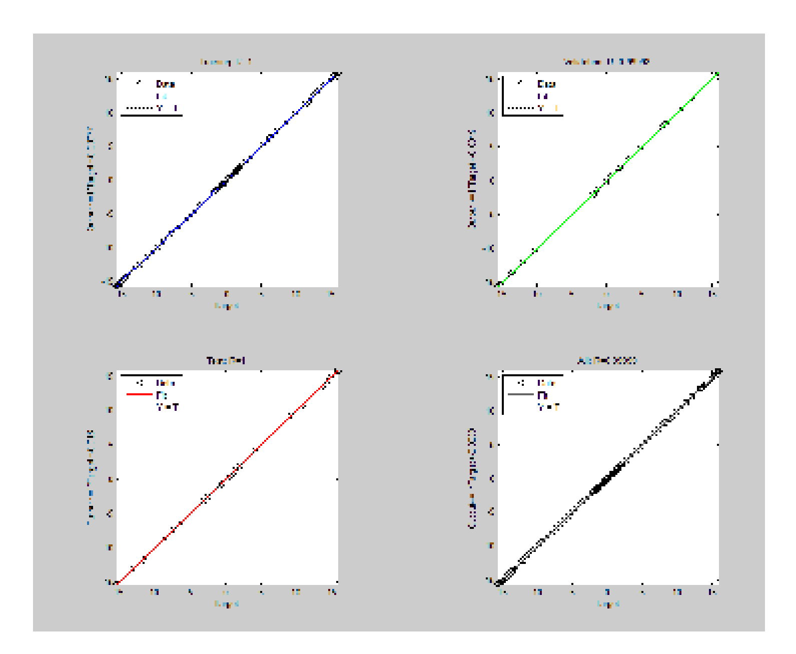


Figure (7) : The Regression of the neural network

Returning to our robotic manipulator , we can input any value of joints' angles (within their physical range of designed motion) starting from theta1 to theta 5 , to deliver the robotic end-effector coordinates x, y, & z by D-H method according to eq.(7) , (8), & (9) alternatively . This what we call Target , while entering the same angles' values to the designed NN will deliver what we call the output , the difference between targets & outputs should be as low as possible connecting them by R as mentioned above, but there will be often this difference as seen by fig.(7),(10) & (11), where not all of the actual outputs' elements will be equal to the targets.

In fact , the best R we had got from our NN was 0.9996 which is so near from the ideal one 1(after more than 115 iterations) . As we reach this value , we can input any joints' angles , for example , the robot home position (figure 2) (ɵ1= 0 , ɵ2= 0 ,ɵ3= 0, ɵ4= -90 , ɵ5= 0) degrees, in D-H method we have x= 18.0000 , y= 0.0000 & z= 0.0121 , while in NN method we have x= 18.0072, y= 0.0000 , & z= 0.0121048.Where the NN results were so sufficient , keeping in mind the NN simplicity and rapidity .

We notice that having a real robotic manipulator to study the forward kinematics gave us the credibility in results , especially after the joints' axes analysis and parameters finding ,depending on the rotation angles' design and links' lengths , and taking in consideration the possible values range of each joint .Also , there will be the ability to convoy the real – time operation of the robot .

An important point must be considered is that ; each time a neural network is trained , can result in a different solution due to different initial weights' and biases' values and different divisions of data into training, validation, and test sets. As a result, different neural networks trained on the same problem can give different outputs for the same input. To ensure that a neural network of good accuracy has been found ,we retrain several times , as could be noted by the results shown in figures 8 to 11 where random initial weights and biases are considered .

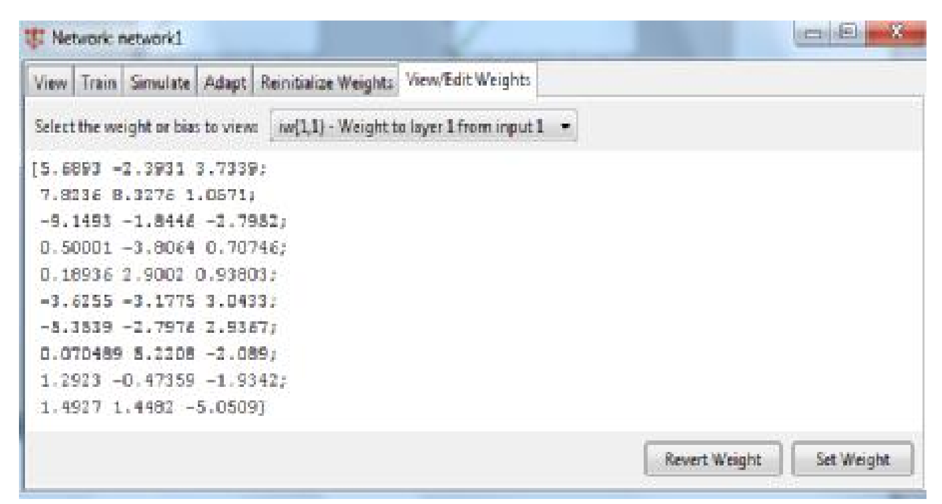


Figure (8) : Selecting the initial weights & biases of layer1 of the neural network

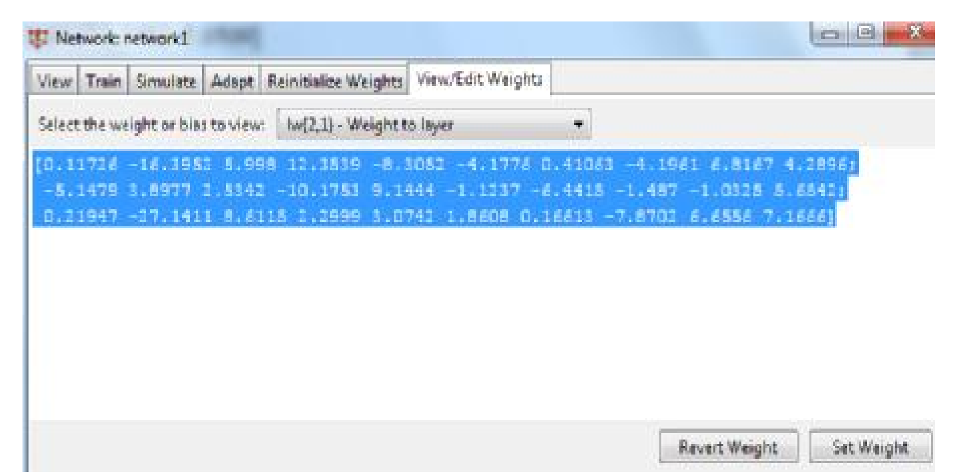


Figure (9) : Selecting the initial weights & biases of layer2 of the neural network

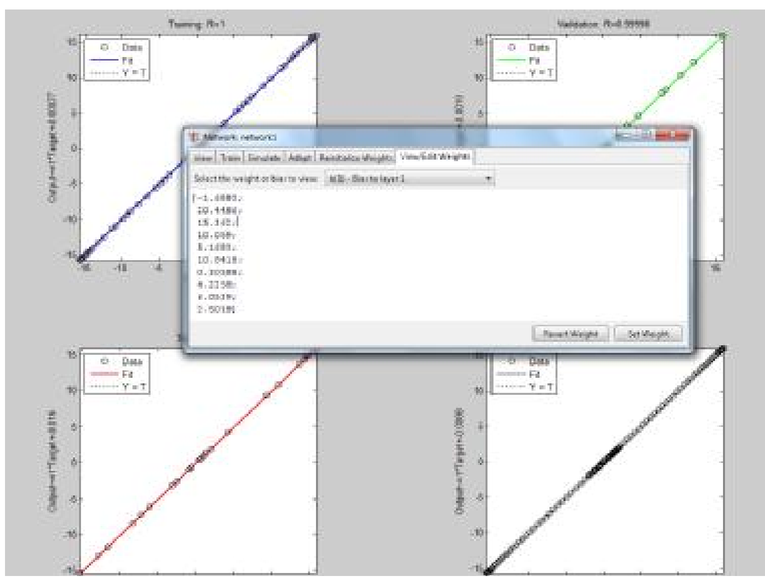


Figure (10) : Regression of NN after changing layer 1 weights & biases

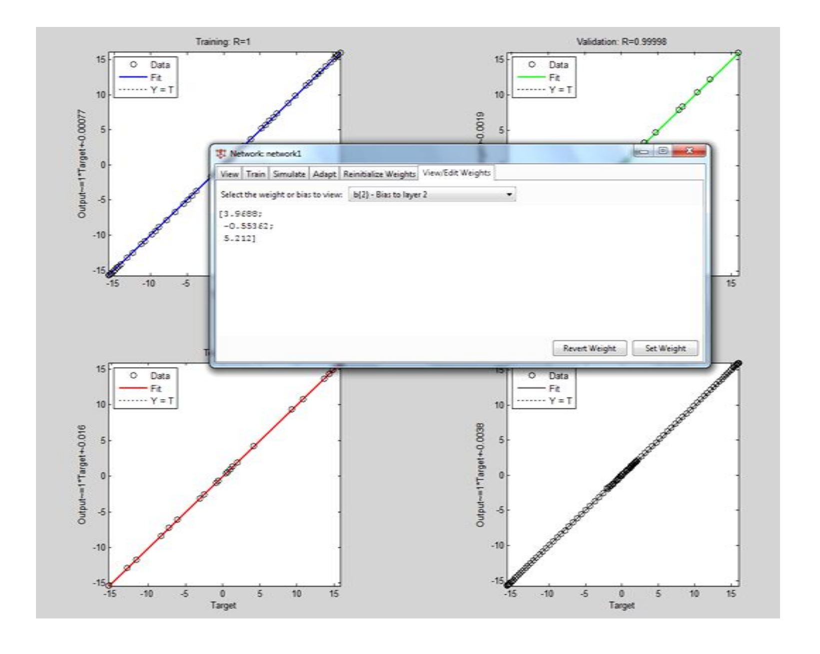


Figure (11) : Regression of NN after changing layer 2 weights & biases

Another note is that as MU controls how much the weights are changed on each iteration , it can be expected that too small of this value will cause the network to converge too slowly. In comparison, too large of this value will cause

the convergence to be erratic, and will exhibit an oscillation around the final solution. Thus , trying a suitable value for this constant may solve the NN difficulty to learn .

Finally , we could see by this study , that NN had offered a simple and fast method to deal with the non linear kinematic equations of the robot , especially as the robot become more complicated with high DOF and noisy data , less time means less effort & cost . Besides , the NN programming is not as complex as the D-H one , with capability to be updated , developed to deal with other robot problems like finding the Inverse Kinematic equations and avoiding the robots' joints singularities .In fact , we can perform a real-time operations on our robot using NN as a development for this work.

On the other hand , the D-H convention method will be sufficient with small robot's DOF , also the NN may have a little less accuracy than D-H . Besides , there must be a skill existence to deal with neural network to reach to the suitable learned one with the most acceptable results . Also , as we notice from figures 5&6 , the learning effect of the neural network has the limitation of falling in remaining of accuracy unchanging after some specified time of learning , this limitation could be decreased by considering some suggestions like changing one or more of the initial NN functions , or changing the number of NN layers or neurons or both of them , or by changing the weights' and biases' values **.**

CONCLUSIONS

In this paper , the forward kinematics analysis of an actual laboratory 5-DOF robotic manipulator was discussed by two parts , the first part had used the Denavit -Hartenberg (D-H) convention method , while the second part had accomplished using the Artificial Neural Network (ANN) method which was simulated by MATLAB programming .

In spite of the importance of the traditional D-H convention method for the sake of analyzing the robot kinematics , a recent universal scientific trend has been appeared to involve the artificial intelligence in the robot motion controlling researches . Thus an artificial neural network had built , by this paper , to overcome the difficulties that occurred in dealing with the kinematics of the nonlinear robotic system . Besides , we noticed that the designed ANN could be trained in a fast manner and acceptable solutions , the differences in actual and predicted target values using this ANN shows that approximately no error was achieved , by other words , the results showed the effectiveness and usefulness of the proposed analysis and motion control of the robotic forward kinematics procedures using ANN.

REFRENCES

[1] Farzin Piltan , Sh. Tayebi Haghighi, N. Sulaiman, I. Nazari & S. Siamak , " Artificial control of PUMA robot manipulator: A review of fuzzy inference engine

and application to classical controller ", International Journal of Robotic and Automation (IJRA), Volume 2 , Issue 5, pp.401, 2011.
[2] Serdar Kucuk and Zafer Bingul , "Robot kinematics: Forward and inverse kinematics" , Industrial-Robotics-Theory-Modeling-Control, pp. 964, Germany, December 2006 .
[3] Jolly Shah, S.S.Rattan, B.C.Nakra , "Kinematic analysis of 2-DOF planer robot using artificial neural network" , World Academy of Science, Engineering and Technology 57, pp.282, 2011.
[4] H. Meshref , "Using the immune network to control a robot arm", pp. 54 , 2002. www.scholar.lib.vt.edu/theses/available/etd ./08_Chapter4.pd.
[5] Scilling J. , Fundamentals Of Robotics Analysis And Control , Prentice Hall of India , 1996.
[6] John M. Hollerbach , "Forward kinematics for position" , pp.1, September 2004.
www.eng.utah.edu/~cs5310/chapter4.
[7] K.Kishore Kumar , Dr.A.Srinath , G.Jugal anvesh , R.Prem sai , M.suresh , "Kinematic analysis and simulation of 6DOF KukaKr5 robot for welding application", International Journal of Engineering Research and Applications (IJERA) , Vol. 3, Issue 2 , pp.820-827 , March -April 2013.
[8] Youshen Xia and Jun Wang , "A dual neural network for kinematic control of redundant robot manipulators" , IEEE Transactions on systems , Man and cybernetics - PART B , Vol. 31, no.1, pp. 147, February 2001 .
[9] Leila Fallah Araghi, M. H. koorayme , "Neural network controller for two links- robotic manipulator control with different load" , Proceedings of the International Multi Conference of Engineers and Computer Scientists 2009 , Vol. II , IMECS , Hong Kong , pp.978, March 18 - 20, 2009.
[10] Omid M. Omidvar and David L. Elliott, Editors , Neural Systems For Control , Academic Press, Elsevier , February, 1997 .
[11] Zhao-Hui Jiang , Hiroshima Institute of Technology , " Trajectory control of robot manipulators using a neural network controller" , Robot Manipulators Trends and Development , pp. 666 , March 2010 .